UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/624,705	07/21/2003	Mark Ronald Plesko	3382-65536	7062

26119 7590 06/15/2009
KLARQUIST SPARKMAN LLP
121 S.W. SALMON STREET
SUITE 1600
PORTLAND, OR 97204

EXAMINER	
DAO, THUY CHAN	
ART UNIT	**PAPER NUMBER**
2192	

MAIL DATE	DELIVERY MODE
06/15/2009	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/624,705	PLESKO ET AL.	
	Examiner	**Art Unit**	
	Thuy Dao	2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *13 April 2009*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1,3-13,15,17-21,23,24,26,28,31-34,36,38 and 39* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1,3-13,15,17-21,23,24,26,28,31-34,36,38 and 39* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *21 July 2003* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)

2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date *04/13 and 05/28/09*.

4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .

5) ☐ Notice of Informal Patent Application

6) ☐ Other: _____.

DETAILED ACTION

1. This action is responsive to the amendment filed on April 13, 2009.

2. Claims 1, 3-13, 15, 17-21, 23, 24, 26, 28, 31-34, 36, 38, and 39 have been examined.

Response to Amendments

3. In the instant amendment, claims 1, 20, 28, 31, 36, 38, and 39 have been amended.

4. The 35 USC §101 rejection over claims 20-24, 26, 28, 36, 38, and 39 is withdrawn in view of Applicant's amendments.

Response to Arguments

5. Applicants' arguments have been considered.

a) Claims 1, 3-13, 15, and 17-19 (Remarks, pp. 9-11):

After further consideration, the examiner notes that Gordon (art of record, US Patent No. 6,560,774) also teaches the newly added limitations. Gordon explicitly teaches:

wherein one of the sub-classes representing a primitive type represents an unknown types (e.g., col.7: 57-65; col.27: 64 - col.28: 7)

wherein the unknown type can represent any type (e.g., col.27: 64 – col.28: 7, field, stack frame offset, and platform architecture not known before compile time) *and*

wherein a compiler drops type information by changing a known type to the unknown type (e.g., col.27: 64 - col.28: 7, type information is deferred until JIT compilation)

during a stage of lowering (e.g., col.1: 12-23 and FIG. 2, col.6: 8-33, transforming source code in high level language to intermediate/low level language, i.e., lowering).

b) Claims 20, 21, 23, 24, 26, and 28 (Remarks, pp. 11-12):

Gordon explicitly teaches:

wherein the class 'PrimType' represents a plurality of types (e.g., col.1: 54-63: native size primitive types; col.7: 66 – col.8: 6: integers, floating points)

comprising at least an unknown type (e.g., col.7: 57-65; col.27: 64 - col.28: 7)

wherein the unknown type can represent any type (e.g., col.27: 64 – col.28: 7) field, stack frame offset, and platform architecture not known before compile time) *and*

wherein a compiler drops type information by changing a known type to the unknown type (e.g., col.27: 64 - col.28: 7, type information is deferred until JIT compilation)

during a stage of lowering (e.g., col.1: 12-23 and FIG. 2, col.6: 8-33, transforming/lowering source code in high level language to intermediate and/or low level language).

c) Claims 31-34 (Remarks, pp. 12-13):

Claim 31 has been amended, in part, similarly to claim 1. Therefore, as the reference teaches all of the limitations of the above claim 1, it also teaches all of the limitations of claim 31. Dependent claims 32-34 are also rejected based on virtue of their dependencies on the rejected base claim 31.

d) Claims 36, 38, and 39 (Remarks, pp. 13-14):

Claim 36 has been amended, in part, similarly to claim 1. Therefore, as the reference teaches all of the limitations of the above claim 1, it also teaches all of the limitations of claim 36. Dependent claims 38 and 39 are also rejected based on virtue of their dependencies on the rejected base claim 36.

In conclusion, the examiner respectfully maintains ground of the 35 USC §102 rejection over claims 1, 3-13, 15, 17-21, 23, 24, 26, 28, 31-34, 36, 38, and 39.

Claim Rejections – 35 USC § 103

6. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious

> at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

7. Claims 1, 3-13, 15, 17-21, 23, 24, 26, 28, and 31-34 are rejected under 35 U.S.C. 103(a) as being unpatentable over Gordon (art of record, US Patent No. 6,560,774) in view of Microsoft-IL (art of record, “Inside Microsoft .NET IL Assembler”).

Claim 1:

Gordon discloses a method of representing type information for a typed intermediate language (see at least Type Definitions col.16:58-col.17:51)

via objects of classes in a class hierarchy (see at least class hierarchies, subtype col.16:33-41),

wherein the class hierarchy comprises at least one class and a plurality of sub-classes for representing different type classifications (see at least class hierarchies, subtype col.16:33-41), the *method comprising:*

instantiating one or more objects of one or more of the sub-classes of the hierarchy (see at least class constructor, initialization col.12:32-34; object constructors col.14:25-57; object initialization col.15:32-34; FIG.12 & associated text),

wherein the one or more sub-classes represent classifications of types for the typed intermediate language (see at least Type Definitions col.16:58-col.17:51; 202 FIG.2 & associated text; 302 FIG.3 & associated text); *and*

storing information in the one or more objects wherein the one or more objects (see at least Field Definitions in Types, Method Definitions in Types col.17:43-col.18:10) *wherein the typed intermediate language is capable of representing a plurality of different programming languages* (see at least FIG.3 & associated text); *and*

wherein the one or more objects represent type information for instructions in the typed intermediate language (see at least FIG.18 & associated text);

wherein one of the sub-classes representing a primitive type represents an unknown types (e.g., col.7: 57-65; col.27: 64 - col.28: 7)

wherein the unknown type can represent any type (e.g., col.27: 64 – col.28: 7) field, stack frame offset, and platform architecture not known before compile time) *and*

wherein a compiler drops type information by changing a known type to the unknown type (e.g., col.27: 64 - col.28: 7, type information is deferred until JIT compilation)

during a stage of lowering (e.g., col.1: 12-23 and FIG. 2, col.6: 8-33, transforming/lowering source code in high level language to intermediate and/or low level language).

Gordon does not explicitly disclose *the classifications of types comprises a primitive type associated with a primitive size, and wherein the primitive type size is settable to represent a constant size, the primitive type size is set-table to represent a symbolic size, and the primitive type size is settable to represent an unknown size.*

However, Microsoft-IL further discloses:

the classifications of types comprises a primitive type associated with a primitive type size (e.g., chapter 7, pp. 1-2)*, and*

wherein the primitive type size is settable to represent a constant size (e.g., chapter 7, page 2, Table 7-1, primitive type sizes such as unsigned 2-byte integer, 4-byte floating-point to present constant sizes 2 bytes and/or 4 bytes)

the primitive type size is set-table to represent a symbolic size (e.g., chapter 7, page 2, Table 7-1, primitive type BOOLEAN has a symbolic size Byte, primitive type SByte has a symbolic size Byte, and primitive type Byte has a symbolic size Byte)*, and*

the primitive type size is settable to represent an unknown size (e.g., chapter 7, page 2, Table 7-1, primitive type IntPtr has an unknown size, which is dependent on the underlying platform).

It would have been obvious to a person having ordinary skill in the art at the time the invention was made to combine Microsoft-IL's teaching into Gordon's teaching. One

would have been motivated to do so to define types for an intermediate language in the .NET Framework class library as suggested by Microsoft-IL (e.g., chapter 7, page 1).

Claim 3:

The rejection of base claim 1 is incorporated. Gordon discloses *wherein at least one of the objects comprises information for a size of a type represented by the object* (see at least FIG.24 & associated text).

Claim 4:

The rejection of base claim 1 is incorporated. Gordon discloses *wherein at least one of the one or more sub-classes inherits from an abstract type that wraps an externally defined type, the abstract type providing a mapping from the typed intermediate language to original source code* (see at least programs, objects, components, data structures, abstract data types col.4:15-22; class, abstract, superclass col.17:9-14).

Claim 5:

The rejection of base claim 1 is incorporated. Gordon discloses *wherein at least one of the one or more sub-classes represents container types* (see at least FIG.15 & associated text; FIG.26 & associated text).

Claim 6:

The rejection of base claim 1 is incorporated. Gordon discloses *wherein at least one of the one or more sub-classes represents pointer types* (see at least FIG.11 & associated text; FIG.12 & associated text; *IL Instructions and Pointer Types* col.29:60-col.30:25).

Claim 7:

The rejection of base claim 1 is incorporated. Gordon discloses *wherein at least one of the one or more sub-classes represents function types* (see at least FIG.22 & associated text; *Method Definitions in Types* col.17:52-col.18:10).

Claim 8:

The rejection of base claim 1 is incorporated. Gordon discloses *wherein at least one of the one or more sub-classes represents unmanaged array types* (see at least FIG.26 & associated text; FIG.15 & associated text).

Claim 9:

The rejection of base claim 1 is incorporated. Gordon discloses *wherein at least one of the one or more sub-classes represents class types* (see at least *Type Definitions, type, class* col.16:58-65).

Claim 10:

The rejection of base claim 1 is incorporated. Gordon discloses *wherein at least one of the one or more sub-classes represents managed array types* (see at least FIG.26 & associated text; FIG.15 & associated text).

Claim 11:

The rejection of base claim 1 is incorporated. Gordon discloses *wherein at least one of the one or more sub-classes represents struct types* (see at least FIG.26 & associated text; FIG.15 & associated text).

Claim 12:

The rejection of base claim 1 is incorporated. Gordon discloses *wherein at least one of the one or more sub-classes represents interface types* (see at least *Type Definitions, interface* col.16:58-65).

Claim 13:

The rejection of base claim 1 is incorporated. Gordon discloses *wherein at least one of the one or more sub-classes represents enumerated types* (see at least *Type Definitions, enumeration* col.16:58-65).

Claim 14:

The rejection of base claim 1 is incorporated. Gordon discloses *wherein at least one of the one or more sub-classes represents primitive types* (see at least FIG.6 & associated text; FIG.27 & associated text).

Claim 15:

The rejection of claim 1 is incorporated. Gordon discloses *wherein at least one of the sub-classes representing primitive types represents the following types: int, float, and void* (see at least FIG.6 & associated text; FIG.27 & associated text).

Claim 16:

The rejection of claim 1 is incorporated. Gordon discloses *wherein at least one of the sub-classes representing primitive types can represent an unknown type* (see at least *unboxed* col.30:25-45).

Claim 17:

The rejection of claim 1 is incorporated. Gordon discloses *wherein at least one of the sub-classes representing primitive types is extensible to represent one or more additional primitive types* (see at least FIG.6 & associated text; FIG.27 & associated text).

Claim 18:

The rejection of claim 1 is incorporated. Microsoft-IL discloses *at least one of the one or more sub-classes is defined from the group consisting of: 'ContainerType', 'PtrType', 'FuncType', 'ClassType', 'StructType', 'InterfaceType', and' EnumType'* (e.g., chapter 6, Interfaces and Enumerators; chapter 7, pp. 1-4, Table 7-1 and page 10).

Claim 19:

The rejection of claim 1 is incorporated. Gordon discloses *at least one of the one or more sub- classes is defined as 'PrimType'* (e.g., col.4: 15-22; col.17: 9-14).

Claim 20:

Claim 20 is a computer-readable medium version, which recites the same limitations as those of claim 1, wherein all claimed limitations have been addressed and/or set forth above. Therefore, as the references teach all of the limitations of the above claim 1, they also teach all of the limitations of claim 20.

Claim 21:

The rejection of claim 20 is incorporated. Gordon discloses *the size represents a size of a machine representation of a value* (e.g., col.16: 58 – col.17: 51; col.12: 32-34).

Claim 22:

The rejection of claim 20 is incorporated. Gordon discloses *associating a size with instances of the 'PrimType' class comprises defining the size as 'BitSize'* (e.g., col.17: 42 – col.18: 10; FIG. 12 and associated text).

Claim 23:

The rejection of claim 20 is incorporated. Gordon discloses *the kind of type represents a type classification* (e.g., col.14: 25-57; col.16: 33-41).

Claim 24:

The rejection of claim 20 is incorporated. Gordon discloses *associating a kind of primitive type with instances of the 'PrimType' class comprises defining the kind of type as 'PrimTypekind'* (e.g., col.4: 15-22; FIG. 18 and associated text).

Claim 26:

The rejection of claim 20 is incorporated. Gordon discloses *associating a type of size with instances of the 'PrimType' class comprises defining the type of size as 'SizeKind'* (e.g., col.29: 60 – col.30: 25; col.17: 52 – col.18: 10).

Claim 28:

The rejection of claim 20 is incorporated. Gordon discloses *the class 'PrimType' represents a plurality of types, the plurality of types comprising int, float, unknown, void, condition code, and unsigned int types* (e.g., col.4: 15-22; col.17: 9-14; FIG. 15, FIG. 26 and associated text).

Claim 31:

Claim 31 is a method version, which recites the same limitations as those of claim 1, wherein all claimed limitations have been addressed and/or set forth above. Therefore, as the references teach all of the limitations of the above claim 1, they also teach all of the limitations of claim 31.

Claim 32:

The rejection of claim 31 is incorporated. Microsoft-IL discloses *defining a plurality of classes hierarchically below the class representing container types, wherein the plurality of classes represent type information for the typed intermediate language, and wherein the plurality of classes represent at least class types, struct types, interface types, and enumerated types of a plurality of programming languages* (e.g., chapter 7, pp. 1-4, Table 7-1 and 7-6).

Claim 33:

The rejection of claim 32 is incorporated. Gordon discloses *defining a class hierarchically below the class representing class types, wherein the class represents type information for the typed intermediate language, and wherein the class represents unmanaged array types of a plurality of programming languages* (e.g., col.29: 60 – col.30: 25; col.17: 52 – col.18: 10).

Claim 34:

The rejection of claim 31 is incorporated. Gordon discloses *defining a class hierarchically below one of the plurality of classes, wherein the class represents type information for the typed intermediate language* (e.g., col.16: 58-65; FIG. 15 and associated text).

Claim 35:

The rejection of claim 34 is incorporated. Gordon discloses *the class further represents an unknown type* (e.g., col.30: 25-45; FIG. 6, FIG. 27 and associated text).

8. Claims 36, 38 and 39 are rejected under 35 U.S.C. 103(a) as being unpatentable over Microsoft-IL in view of Syme (art of record, US Patent No. 7,346,901) and Gordon.

Claim 36:

Microsoft-IL discloses *a computer-readable medium having a software program thereon, the program comprising computer executable instructions for implementing a method for representing type information for a typed intermediate language using a class hierarchy for representing different type classifications, the method comprising:*

defining a programming class of the class hierarchy as 'PtrType', wherein an object of class 'PtrType' is a type representation for the typed intermediate language for pointer types in a section of code written in one of a plurality of programming languages (e.g., chapter 7, page 2, Table 7-1, integer pointer type and unsigned integer pointer type; pp. 3-4, data pointer types)*;*

defining a programming class of the class hierarchy as 'FuncType', wherein an object of class 'FuncType' is a type representation for the typed intermediate language for function types in a section of code written in one of a plurality of programming languages (e.g., pp. 4-5, function pointer types which point to functions)*;*

defining a programming class of the class hierarchy as 'ClassType', wherein an object of class 'ClassType' is a type representation for the typed

intermediate language for class types in a section of code written in one of a plurality of programming languages (e.g., pp. 5-6, vectors and arrays are class instances derived from the abstract class [mscorlib]System.Array written in one of a plurality of programming languages)*;*

defining a programming class of the class hierarchy as 'StructType', wherein an object of class ' StructType' is a type representation for the typed intermediate language for struct types in a section of code written in one of a plurality of programming languages (e.g., page 10, Struct)*;*

defining a programming class of the class hierarchy as 'InterfaceType', wherein an object of class 'InterfaceType' is a type representation for the typed intermediate language for interface types in a section of code written in one of a plurality of programming languages (e.g., chapter 6, Namespaces and Classes > Interfaces)*; and*

defining a programming class of the class hierarchy as 'EnumType', wherein an object of class 'EnumType' is a type representation for the typed intermediate language for enumerated types in a section of code written in one of a plurality of programming languages (e.g., chapter 6, Namespaces and Classes > Enumerators)*; and*

defining a programming class of the class hierarchy as 'PrimType', wherein an object of class 'PrimType' is a type representation for the typed intermediate language for primitive types in a section of code written in one of a plurality of programming languages; wherein the object of class 'PrimType' is associated with a size settable to represent a constant size, settable to represent a symbolic size, and settable to represent an unknown size (e.g., chapter 7, Primitive Types and Signatures > Primitive Types in the Common Language Runtime)*;*

the object of class 'PrimType' is associated with a size settable to represent a constant size for the object of class 'PrimType' (e.g., chapter 7, page 2, Table 7-1, primitive type sizes such as unsigned 2-byte integer, 4-byte floating-point to present constant sizes)*,*

set-table to represent a symbolic size for the object of class 'PrimType' (e.g., chapter 7, page 2, Table 7-1, primitive type BOOLEAN has a symbolic size Byte, primitive type SByte has a symbolic size Byte, and primitive type Byte has a symbolic size Byte), *and*

settable to represent an unknown size for the object of class 'PrimType' (e.g., chapter 7, page 2, Table 7-1, primitive type IntPtr has an unknown size, which is dependent on the underlying platform).

Microsoft-IL does not explicitly disclose *defining a programming class of the class hierarchy as 'ContainerType', wherein an object of class 'ContainerType' is a type representation for the typed intermediate language for container types in a section of code written in one of a plurality of programming languages.*

However, in an analogous art, Syme further discloses *defining a programming class of the class hierarchy as 'ContainerType', wherein an object of class 'ContainerType' is a type representation for the typed intermediate language for container types in a section of code written in one of a plurality of programming languages* (e.g., col.10: 27-67).

It would have been obvious to a person having ordinary skill in the art at the time the invention was made to combine Syme's teaching into Microsoft-IL's teaching. One would have been motivated to do so to contain generic code in the container class for efficient execution (e.g., col.10: 37-43; col.1: 6-9; col.2: 4-17).

Neither Microsoft-IL nor Syme explicitly discloses the remained limitations. However, in an analogous art, Gordon further teaches:

one of the sub-classes representing a primitive type represents an unknown types (e.g., col.7: 57-65; col.27: 64 - col.28: 7)

wherein the unknown type can represent any type (e.g., col.27: 64 – col.28: 7) field, stack frame offset, and platform architecture not known before compile time) *and*

wherein a compiler drops type information by changing a known type to the unknown type (e.g., col.27: 64 - col.28: 7, type information is deferred until JIT compilation)

during a stage of lowering (e.g., col.1: 12-23 and FIG. 2, col.6: 8-33, transforming/lowering source code in high level language to intermediate and/or low level language).

It would have been obvious to a person having ordinary skill in the art at the time the invention was made to combine Gordon's teaching into Microsoft-IL and Syme's teaching. One would have been motivated to do so have class type independent with platform architecture and type information deferred until JIT compilation as suggested by Gordon (e.g., col.27: 64 – col.28: 7).

Claim 38:

The rejection of claim 36 is incorporated. Microsoft-IL discloses *further comprises program code for associating a size with an object of any class* (e.g., chapter 7, page 2, Table 7-1).

Claim 39:

The rejection of claim 36 is incorporated. Microsoft-IL discloses *further comprises program code for associating a kind of type with an object of any class* (e.g., pp. 9-11, Table 7-6).

Conclusion

9. Applicants' amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the

shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

10. Any inquiry concerning this communication should be directed to examiner Thuy Dao (Twee), whose telephone/fax numbers are (571) 272 8570 and (571) 273 8570, respectively. The examiner can normally be reached on every Tuesday, Thursday, and Friday from 6:00AM to 6:00PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam, can be reached at (571) 272 3695.

The fax phone number for the organization where this application or proceeding is assigned is (571) 273 8300.

Any inquiry of a general nature of relating to the status of this application or proceeding should be directed to the TC 2100 Group receptionist whose telephone number is (571) 272 2100.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

/Thuy Dao/
Examiner, Art Unit 2192

/Tuan Q. Dam/
Supervisory Patent Examiner, Art Unit 2192